1 Hashing: Recap

We saw the ideas of universal hash families and k-wise independent hash families in the previous lecture: let us recap them.

Definition 1 *A family H of hash functions mapping U to $[M]$ is called universal if for any two keys $x \neq y \in U$, we have*

$$\Pr_{h \leftarrow H}\left[h(x) = h(y)\right] \leq \frac{1}{M}.$$

Make sure you understand the definition. This condition must hold for *every pair* of distinct keys, and the randomness is over the choice of the actual hash function h from the set H.

Definition 2 *A family H of hash functions mapping U to $[M]$ is called k-wise-independent if for any k distinct keys $x_1, x_2, \ldots, x_k \in U$, and any k values $\alpha_1, \alpha_2, \ldots, \alpha_k \in [M]$ (not necessarily distinct), we have*

$$\Pr_{h \leftarrow H}\left[h(x_1) = \alpha_1 \ \wedge \ h(x_2) = \alpha_2 \ \wedge \ \cdots \ \wedge \ h(x_k) = \alpha_k\right] \leq \frac{1}{M^k}.$$

Such a hash family is also called k-wise independent. The case $k = 2$ is called pairwise independent.

2 Load Balancing

Another central application of hashing is in load balancing. Suppose there are N jobs to be sent to M machines, and consider the case where $M = N$. So there exists a way to send each job to a machine and maintain load 1. But if we hash the jobs to machines, we will get some additional load due to randomness. How much?

Let's use the same formalism as last time.

- Jobs are indexed by keys from the universe U, and we have a set S of $|S| = N$ jobs to schedule. Let us imagine that each job has the same (unit) size.
- There are M machines, indexed by $[M] = \{0, 1, 2, \ldots, M-1\}$.
- We have a family H of hash functions $\{h_1, h_2, \ldots, h_k\}$, with each $h_i : U \to [M]$. We randomly pick a hash function $h \leftarrow H$, and then each job $x \in U$ is mapped to machine $h(x)$.

We want to analyze the "load" of this strategy. It is clear that the best way to schedule N jobs on M machines is to assign N/M jobs to each machine. Can we show that there exist hash families H such that for every subset set S of jobs, the load on all machines is $\approx N/M$ with high probability? Let's see.

> *Notation: We will often call jobs as "balls" and machines as "bins". Think of throwing balls into bins. You want no bin to get many balls. The "load" of a bin is the number of balls that map to it.*

2.1 Load-Balancing Using Hashing

To begin, consider the simplest case of $N = M$. We would like each machine to have $N/M = 1$ jobs, the average load. Suppose the hash functions were truly random: each $x \in U$ was mapped independently to a random machine in $[M]$. What is the maximum load in that case? Surprisingly, you can show:

Theorem 3 *The max-loaded bin has $O(\frac{\log N}{\log \log N})$ balls with probability at least $1 - 1/N$.*

PROOF: The proof is a simple counting argument. The details are not important for this course — the main point is the idea, which says (a) show that the expected number of bins with more than k balls is $1/N$, so the probability that we have one bin with more than k balls is N times more than the expectation, which by Markov's is at most $1/N$.

Here are the details, for completeness. The probability that some particular bin i has at least k balls is at most

$$\binom{N}{k}\left(\frac{1}{N}\right)^k \leq \frac{N^k}{k!} \cdot \frac{1}{N^k} \leq \frac{1}{k!} \leq 1/k^{k/2}$$

which is (say) $\leq 1/N^2$ for $k = \frac{8 \log N}{\log \log N}$. To see this, note that

$$k^{k/2} \geq (\sqrt{\log N})^{4 \log N / \log \log N} \geq 2^{2 \log N} = N^2.$$

So union bounding over all the bins, the chance of some bin having more than $8\frac{\log N}{\log \log N}$ balls is $1/N$. (I've been sloppy with constants, you can get better constants using Stirling's approximation.) □

Moreover, you can show that *this is tight*. The max-load with $M = N$ is at least $\Omega(\frac{\log N}{\log \log N})$ with high probability—we are not showing this proof here, but you can check out the proof here if you want. So even with truly random hash functions, the load is much above the average.

Observe that the calculation showing that the maximum load is $O(\frac{\log N}{\log \log N})$ only used that every set of $O(\frac{\log N}{\log \log N})$ balls behaves independently. This means that we do not need the hash family to be fully independent: it suffices to use $O(\frac{\log N}{\log \log N})$-wise-independent hash family to assign balls to bins.

Still, storing and computing with $O(\frac{\log N}{\log \log N})$-wise-independent hash families is expensive. What happens if we use universal hash families to map balls to bins? Or k-wise-independent for $k \ll \frac{\log N}{\log \log N}$? How does the maximum load change? For that it will be useful to look at some *concentration* bounds.

2.2 Concentration Bounds

What is a concentration bound? Loosely, you want to say that some random variable stays "close to" its expectation "most of the time".

2.2.1 Markov's Inequality: Using the Expectation

The most basic bound is *Markov's inequality*, which says that any non-negative random variable is "not much higher" than its expectation with "reasonable" probability. If X is a non-negative random variable, then

$$\Pr[\ X \geq kE[X]\ ] \leq \frac{1}{k}.$$

This bound seems trivial, but all the following bounds come from taking Markov's inequality and applying it to some function of X.

2.2.2 Chebyshev's Inequality: Using the Variance

Using the second moment (this is $E[X^2]$) of the random variable, we can say something stronger. Define $\text{Var}(X) = \sigma^2 = (E[X^2] - E[X]^2)$, the variance of the random variable.

$$\Pr[\ |X - E[X]| \geq k\sigma\ ] \leq \frac{1}{k^2}.$$

Now we don't need X to be non-negative. This is *Chebyshev's inequality.* The proof, interestingly, just applies Markov's to the r.v. $Y = (X - E[X])^2$.

Example: Let's get some practice with using Chebyshev's inequality:

Lemma 4 *Suppose you map N balls into $M = N$ bins using a 2-wise-independent hash family H, maximum load over all bins is $O(\sqrt{N})$ with probability at least $1/2$.*

PROOF: Let L_i be the load of bin i. Let X_{ij} be the indicator random variable for whether ball j fell into bin i. Note that $E[X_{ij}] = 1/M$, and hence $E[L_i] = \sum_{j=1}^{N} E[X_{ij}] = N/M$. Moreover, since the variables are pairwise-independent, we have $\text{Var}(L_i) = \sum_{j=1}^{N} \text{Var}(X_{ij})$. (Verify this for yourself!) And $\text{Var}(X_{ij}) = E[X_{ij}^2] - E[X_{ij}]^2 = E[X_{ij}] - E[X_{ij}]^2 = (1/M - 1/M^2)$. So $\text{Var}(L_i) = N(\frac{1}{M} - \frac{1}{M}^2)$.

Now Chebyshev says the probability of deviation $|L_i - N/M|$ being more than $\sqrt{2M} \cdot \sqrt{\text{Var}(L_i)} \leq \sqrt{2N}$ is at most $\frac{1}{2M}$. And taking a union bound over all M bins means that with probability at least half, all bins have at most $N/M + \sqrt{2N}$ balls. □

Hmm. If we use 2-wise-independent hash families to throw N balls into N bins, we are guaranteed a maximum load $O(\sqrt{N})$ when $N = M$. (For this proof we used that any two balls behaved uniformly and independently of each other.) But using fully random hash functions — or even $O(\frac{\log N}{\log \log N})$-wise-independent hash functions — gives us max-load $\Theta(\frac{\log N}{\log \log N})$.

How does the max-load change when we increase the independence from 2 to fully random? For this, let us give better concentration bounds, which use information about the higher moments of the random variables.

2.2.3 Higher-Moment Chebyshev

A *higher-moment Chebyshev* shows that for any random variable X and even powers $p \geq 2$,

$$\Pr[\ |X - E[X]| \geq D\ ] \leq \frac{E[(X - E[X])^p]}{D^p}.$$

You can use this to show better bounds for hashing using p-wise-independent hash families.

2.2.4 Chernoff/Hoeffding Concentration Bounds

Perhaps the most useful concentration bound is when X is the sum of *bounded independent* random variables. Here we will see Hoeffding's bound that is very useful and easy-to-use, though it's not the most powerful.

Before we give this, let us give some context. Recall the Central Limit Theorem (CLT): it says that if we take a large number of copies $X_1, X_2, \ldots, X_n$ of some independent random variable with mean μ and variance σ^2, then their average behaves like a standard Normal r.v. (a.k.a. Gaussian random variable) in the limit; i.e.,

$$\lim_{n\to\infty} \frac{(\sum_i X_i)/n - \mu}{\sigma^2} \sim N(0,1).$$

And the standard Normal is very concentrated: recall that its density function is

$$\frac{1}{\sqrt{2\pi}\sigma} e^{-\|x-\mu\|^2/2\sigma^2},$$

and the density falls off very rapidly as we get away from the mean. In fact the probability of being outside $\mu \pm k\sigma$, i.e., being k standard deviations away from the mean, drops *exponentially* in k. You should think of the Hoeffding bound below as one quantitative version of the CLT.

Theorem 5 (Hoeffding's Bound) *Suppose $X = X_1 + X_2 + \ldots + X_n$, where the X_is are independent random variables taking on values in the interval $[0,1]$. Let $\mu = E[X] = \sum_i E[X_i]$. Then*

$$\Pr[X > \mu + \lambda] \leq \exp\left(-\frac{\lambda^2}{2\mu + \lambda}\right) \tag{1}$$

$$\Pr[X < \mu - \lambda] \leq \exp\left(-\frac{\lambda^2}{3\mu}\right) \tag{2}$$

A comment on Hoeffding's bound.[1] Suppose $\lambda = c\mu$. Then we see that the probability of deviating by $c\mu$ drops *exponentially* in c. Compare this to Markov's or Chebyshev's, which only give an inverse polynomial dependence ($1/c$ and $1/c^2$ respectively).

> **Example:** Suppose each X_i is either 0 with probability 1/2, or 1 with probability 1/2. (Such an X_i is called a Bernoulli r.v.) Let $X = \sum_{i=1}^n X_i$. If you think of 1 has "heads" and 0 as "tails" then X is the number of heads in a sequence of n coin flips. If n is large, by the Central Limit Theorem we expect this number to be very close to the average. Let's see how we get this.
>
> Each $E[X_i] = 1/2$, hence $\mu := E[X] = n/2$. The bound (1) above says that
>
> $$\Pr[X > n/2 + \lambda] \leq \exp\left(-\frac{\lambda^2}{2(n/2) + \lambda}\right)$$
>
> Clearly $\lambda > n/2$ is not interesting (for such large λ, the probability is zero), so focus on $\lambda \leq n/2$. In this case $2(n/2) + \lambda \leq 3(n/2)$, so the RHS above is at most $e^{-\lambda^2/(3n/2)}$.
>
> E.g., for $\lambda = 30\sqrt{n}$, the probability of $X \geq n/2 + 30\sqrt{n}$ is at most $e^{-(900n)/(3n/2)} = e^{-600}$. (Smaller than current estimates of the number of particles in the universe.) A similar calculation using (2) shows that $\Pr[X < n/2 - \lambda] \leq e^{-\lambda^2/(3n/2)}$. Since the standard-deviation $\sigma = \sqrt{n}$ in this case, these results are qualitatively similar to what the CLT would give in the limit.

[1] Plus a jargon alert: such "exponential" concentration bounds for sums of independent random variables go by the name "Chernoff Bounds" in the computer science literature. It stems from the use of one such bound originally proved by Herman Chernoff. But we will be well-behaved and call these "concentration bounds".

Example: Back to balls-and-bins: let's see how to use Hoeffding's bound to get an estimate of the load of the max-loaded bin.

So, in the $N = M$ case, let L_i be the load on bin i, as above. It is the sum of N i.i.d. random variables X_{ij}, each one taking values in $\{0, 1\}$. We can apply Hoeffding's bound. Here, $\mu = E[X] = N/M = 1$. Set $\lambda = 6 \log N$. Then we get

$$\Pr[L_i > \mu + \lambda] \leq \exp\left(-\frac{\lambda^2}{2\mu + \lambda}\right) \leq \exp\left(-\frac{(6 \log N)^2}{2 + 6 \log N}\right) \leq e^{-2 \log N} = \frac{1}{N^2}.$$

Now we can take a union bound over all the N bins to claim that with probability at least $1 - 1/N$, the maximum load is at most $O(\log N)$. This is weaker than what we showed in Theorem 3, but it still is almost right.[2]

2.2.5 Beware: Need Independence and Boundedness

The price we pay for such a strong concentration are the two requirements that (a) the r.v.s be independent and (b) bounded in $[0, 1]$. We can relax these conditions slightly, but some constraints are needed for sure. E.g., if X_i's are not independent, then you could take the X_i's to be *all equal*, with value 0 w.p. 1/2 and 1 w.p. 1/2. Then $\sum_{i=1}^{n} X_i$ would be either 0 or n each with probability 1/2, and you cannot get the kind of concentration around $n/2$ that we get in the example above.

Similarly, we do need some "boundedness" assumption on the random variables we are summing up. Else imagine that each X_i is independent, but now 0 w.p. $1 - 1/2n$ and $2n$ w.p. $1/2n$. The expectation $E[X_i] = 1$ and hence $\mu = \sum_i E[X_i] = n$. But again you cannot hope to get the kind of concentration given by the Hoeffding bound, since there will be a $1 - (1 - 1/2n)^n \approx 1 - e^{-1/2}$ (which is a constant) probability of the sum being at least $2n$.

3 Load Balancing using Two-Choice Hashing

Just like in cuckoo hashing, here's a more nuanced way to use hashing for load balancing — use two hash functions instead of one! The setting now is: N balls, $M = N$ bins. However, when you consider a ball, you pick *two* bins (or in general, d bins) independently and uniformly at random, and put the ball in the less loaded of the two bins. The main theorem is:

Theorem 6 (Azar, Broder, Karlin, Upfal [ABKU99]) *For any $d \geq 2$, the d-choice process gives a maximum load of*

$$\frac{\ln \ln N}{\ln d} \pm O(1)$$

with probability at least $1 - O(\frac{1}{N})$.

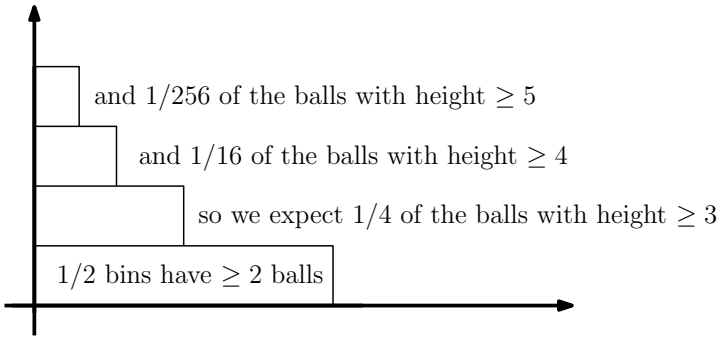
It's pretty amazing: just by looking at two bins instead of one, and choosing the better bin, gives us an exponential improvement on the maximum load: from $\approx \log N$ to $\log \log N$. Moreover, this analysis is tight. (Finally, looking at $d > 2$ bins does not help much further.)

Why is this result important? It is clearly useful for load balancing; if we want to distribute jobs among machines, two hash functions are qualitatively superior to one. We can even use it to give a simple data structure with good worst-case performance: if we hash N keys into a table with $M = N$ bins, but we store up to $O(\log \log N)$ keys in each bin and use two hash functions instead of just one, we can do inserts, deletes and queries in time $O(\log \log N)$.

[2] We stated Hoeffding's bound in its most easy-to-use format. The actual bound is tighter and shows load $O(\frac{\log N}{\log \log N})$. See the last bound on page 2 of these notes if you are interested.

3.1 Some Intuition

The intuition behind the proof is the following picture: Imagine putting the balls in one by one. A bin has height h if it has h balls in it. Say a ball has height h if it is placed in a bin that has $h-1$ balls before this ball was added into it. (We would like to show that no ball has height more than $\frac{\ln \ln N}{\ln 2} + O(1)$.) A bin has height $\geq h$ if it contains a ball of height h.



How many bins can have height (at the end) at least 2? Clearly, at most $N/2$, at most half the bins. (This is just by Markov's inequality.) Now what is the chance that a ball has height at least 3? When it arrives, it must choose two bins, both having height at least 2. That happens with probability at most $\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$. Hence we expect about $\frac{N}{4}$ *balls* to have height at least 3 — and hence the expected number of *bins* of height at least 3 is also at most $N/4 = N/(2^2)$.

Now suppose the actual number of bins of height at least 3 is a the same as its expectation $N/4$. [3] Then using the same argument, we expect at most $N \cdot (1/4)^2 = N/16 = N/(2^{2^2})$ bins to have height at least 4. And $N \cdot (1/16)^2 = N/256 = N^{2^{2^3}}$ bins of height at least 5, and $N/2^{2^{h-2}}$ bins to have height at least h. For $h = \log \log N + 2$, we expect only one ball to have height h.

Of course this is only in expectation, but one can make this intuition formal using concentration bounds, showing that the process does behave (more or less) like we expect. See these notes for a proof.

3.2 Generalizations

It's very interesting to see how the process behaves when we change things a little. Suppose we divide the N bins into d groups of size N/d each. (Imagine there is some arbitrary but fixed ordering on these groups.) Each ball picks one random bin from each group, and goes into the least loaded bin as before. But if there are ties then it chooses the bin in the "earliest" group (according to this ordering on the groups). This subtle change (due to Vöcking [Vöc03]) now gives us load:

$$2\frac{\log \log n}{d} + O(1).$$

Instead of $\log d$ in the denominator, you now get a d. The intuition is again fairly clean. [Draw on the board.]

[3] This is the only part we're being hand-wavy about: formally, we would show using a concentration bound like Chernoff-Hoeffding that the actual number is very close to its expectation with high probability, and proceed using that.

What about the case when $M \neq N$? For the one-choice setting, we saw that the maximum load was $\frac{N}{M} + O(\sqrt{\frac{N \log m}{M}})$ with high probability. It turns out that the d-choice setting gives us load at most

$$\frac{N}{M} + \frac{\log \log M}{\log d} + O(1)$$

with high probability [BCSV06]. So the deviation of the max-load from the average is now independent of the number of balls, which is very desirable!

Finally, supposed you really wanted to get a constant load. One thing to try is: when the i^{th} ball comes in, pick a random bin for it. If this bin has load at most $\lceil i/M \rceil$, put ball i into it. Else pick a random bin again. Clearly, this process will result in a maximum load of $\lceil N/M \rceil + 1$. But how many random bin choices will it do? [BKSS13] shows that at most $O(N)$ random bins will have to be selected. (The problem is that some steps might require a lot of choices. And since we are not using simple hashing schemes, once a ball is placed somewhere it is not clear how to locate it.) This leads is to the next section, where you want to just use two hash functions, but maintain only low load — in fact only one ball per bin!

References

[ABKU99] Yossi Azar, Andrei Z. Broder, Anna R. Karlin, and Eli Upfal. Balanced allocations. *SIAM J. Comput.*, 29(1):180–200, 1999. 5

[BCSV06] Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. Balanced allocations: The heavily loaded case. *SIAM J. Comput.*, 35(6):1350–1385, 2006. 7

[BKSS13] Petra Berenbrink, Kamyar Khodamoradi, Thomas Sauerwald, and Alexandre Stauffer. Balls-into-bins with nearly optimal load distribution. In *SPAA*, pages 326–335, 2013. 7

[Vöc03] Berthold Vöcking. How asymmetry helps load balancing. *J. ACM*, 50(4):568–589, 2003. 6